

Assignment #2
CSCI 201 Spring 2017
4.5% of course grade

For this assignment you will be adding a simple front end GUI to Cinemate. You will be utilizing your knowledge of css, html, and jsp to implement this. You should not need javascript in order to complete this assignment.

Error Checking

To begin, however, you will be adding some error checking to the back end of your application. You will need to catch the following logical errors within the xml file (I would suggest printing the error message to the console for the sake of testing, but further in the instructions, you will see that the message will need to be printed to the GUI):

- There is an event in a user's feed that does not have the action 'Rated' yet has a value in the <rating> field
- There is an event in a user's feed that has the action 'Rated' yet the <rating> field has been omitted
- There are no movie objects found under the <movies> tag
- A user has a non-existing username in their following list
- A movie has a non-existing genre listed
- An event in a user's feed has a non-existing action listed
- Any of the non-String fields have a String value (<rating> for event and movie objects, and <year> for movies)

Searching Expansion

In addition, you will now need to support the ability for a user to search for another user by first name and last name. Unlike the movie searching, there will not be a submenu for this functionality. Instead, the search parameter provided should now return any user with a case insensitive match to the username, first, or last name.

For example, if there is a user with the username 'jeff', a user with the first name 'Jeff' and a user with the last name 'Jeff', all these users should be returned with the search parameter 'JEFF'.

XML Addition

Lastly, there will now be an additional element added to the user's profile which is a URL that maps to their profile picture.

```
<user>
  <username>knguyen</username>
  <password>knguyen</password>
  <fname>Kien</fname>
  <lname>Nguyen</lname>
```

```
<image>http://www-scf.usc.edu/~csci201/images/
kien_nguyen.jpg</image>
<following>
    <username>jmiller</username>
</following>
<feed></feed>
</user>
```

You will want to store this new field as you will need it to display a user's profile picture on the front end.

Front End Creation

The rest of this assignment will be creating the front end of your application. You will still not be making any changes or additions to the application data in any way. Essentially you will be doing almost the exact same user interaction as in Assignment 1, but using a web front end in lieu of the console. For most of the pages, in order to navigate back to a previous page, you will rely on the back button of the browser you are using.

You **will** be graded on the aesthetics of your web design. In order to do this in a fair way, you must try to make your application look as similar to the screenshots you see in these instructions as possible. By that I mean the placement and size of elements, alignment of text, etc., should follow the same structure as shown in the screenshots. However, you can take artistic liberties with the colors, fonts, borders, shadowing, images, etc. In fact, you must use a css file, and you must include **at minimum** some font, color and border choices on each page.

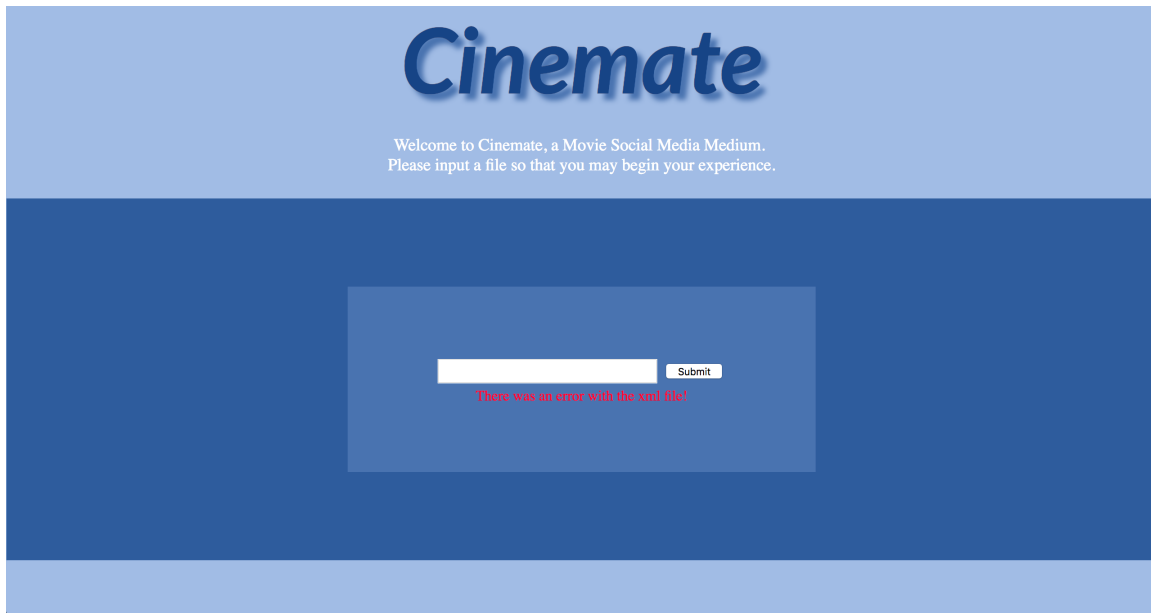
The entry page should look like the following:



Figure 1: Entry page

There should be a large display of the name of the application. Most importantly, there needs to be a text field and a submit button in which the user must enter the **absolute path** to the xml file they wish to parse. If there is an error in the xml file (any of those listed at the beginning of the instructions), an error message must be printed on the page.

Note: In the example below, the error message has been generalized, but you are required to list the exact issue with the file. For example, if there was a movie with a non-existing genre, the message could say “Error! Non-existing genre found in movie object”. If multiple errors are found within the file, only one needs to be displayed (and your choice as to which).



The image shows a web page for 'Cinamate'. At the top, the word 'Cinamate' is displayed in a large, blue, stylized font. Below it, a smaller line of text reads: 'Welcome to Cinamate, a Movie Social Media Medium. Please input a file so that you may begin your experience.' The main content area has a dark blue background. In the center, there is a light blue rectangular box containing a white text input field and a 'Submit' button. Below the input field, a red error message is displayed: 'There was an error with the xml file!'.

Figure 2: Entry Page with Error

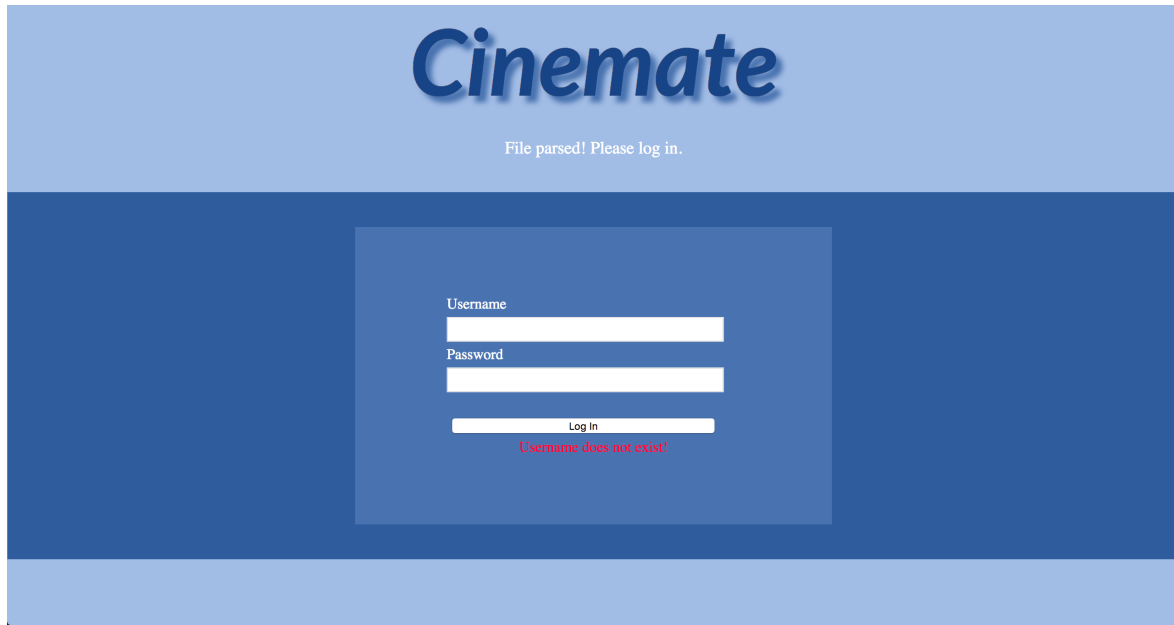
If, however, the file was successfully parsed, the user should be redirected to the start page which should display two text fields and a button. The text fields will be for the username and password, and the button for submitting their credentials to login.



The image shows a web page for 'Cinamate'. At the top, the word 'Cinamate' is displayed in a large, blue, stylized font. Below it, a smaller line of text reads: 'File parsed! Please log in.' The main content area has a dark blue background. In the center, there is a light blue rectangular box containing two white text input fields labeled 'Username' and 'Password', and a 'Log In' button below them.

Figure 3: Start Page

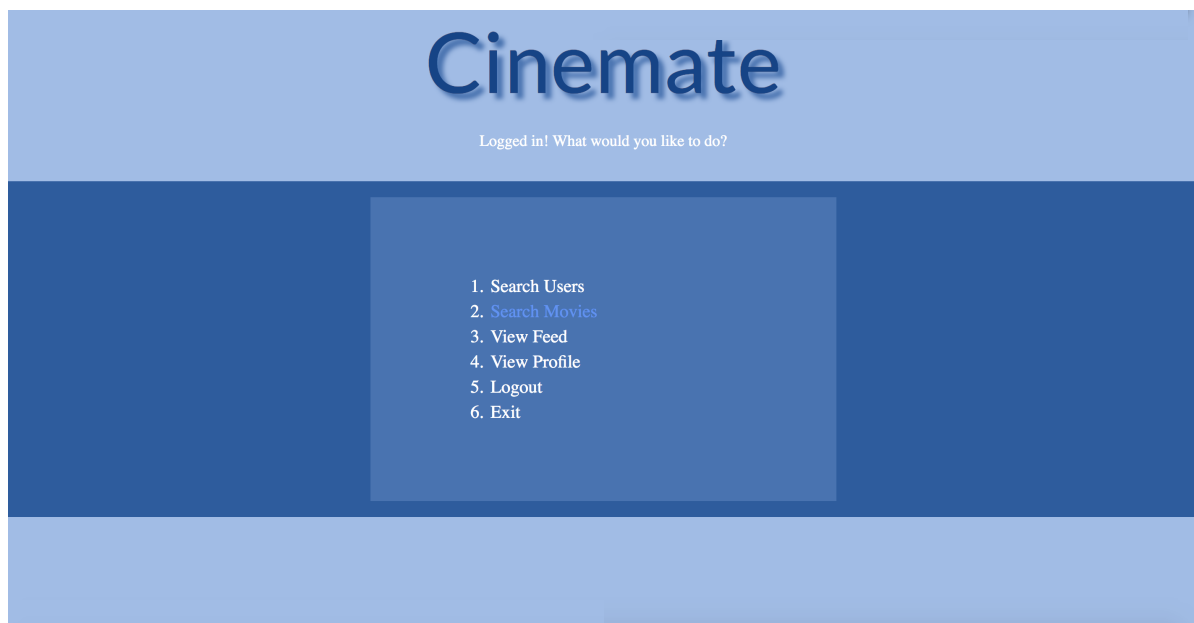
If the username cannot be found, or the password does not match a valid username, the user should remain on this page (it is okay to redirect them to/reload the same page), but an error message should print at the bottom stating the issue. Similar to the entry page, the exact error should be cited (either non-existing username, or incorrect password).



The image shows a web page for 'Cinamate'. At the top, the word 'Cinamate' is displayed in a large, blue, stylized font. Below it, the text 'File parsed! Please log in.' is centered. The main content area has a dark blue background. In the center, there is a white rectangular box containing a login form. The form has two input fields: 'Username' and 'Password'. Below these fields is a 'Log In' button. At the bottom of the form, a red error message reads 'Username does not exist!'.

Figure 4: Start Page with Error

Once the user provides valid credentials, they should be taken to the login page. It will have the same 6 options that they had when they were interacting with the console, but instead of inputting the associated number of the menu option, the menu options should be clickable.



The image shows a web page for 'Cinamate'. At the top, the word 'Cinamate' is displayed in a large, blue, stylized font. Below it, the text 'Logged in! What would you like to do?' is centered. The main content area has a dark blue background. In the center, there is a white rectangular box containing a list of six options: 1. Search Users, 2. Search Movies, 3. View Feed, 4. View Profile, 5. Logout, and 6. Exit. The options are listed in a simple, black, sans-serif font.

Figure 5: Login Page

If the user chooses to log out, they should be redirected back to the start page (Figure 3). If the user chooses to quit, they should be redirected back to the entry page (Figure 1).

If the user chooses to view their profile, they should see the following screen (remember to structure the display of information as closely as you can to the screenshot). This is where you will now be displaying the profile picture.



Figure 6: Profile Page

If the user chooses to view their feed, they should see the following screen. If an event is associated with the logged in user, in this case knguyen, including the username in the event description is optional (like in the screenshot). However, if the event is from a user they are following, the username should be present in the event description just as in Assignment 1.

In addition, adding an associated png for an event (a thumbs up for 'Liked', thumbs down for 'Disliked', etc.) is **optional**, but encouraged!

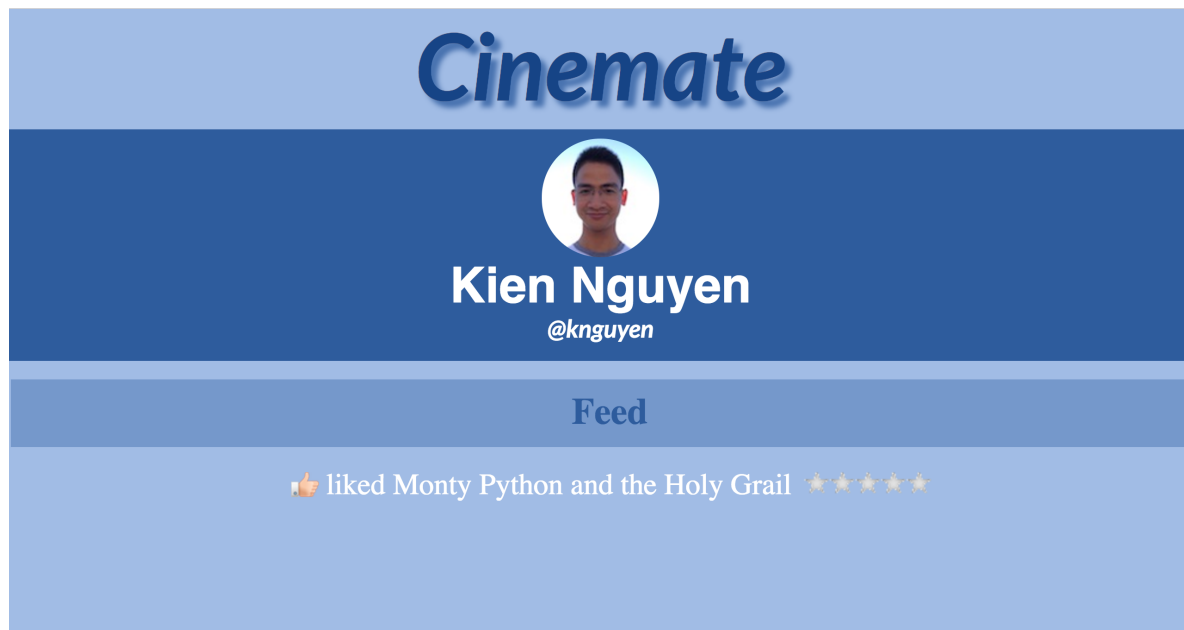


Figure 7: Feed Page

If the user chooses to search for a movie, they should be redirected to a page with the movie searching submenu. It should show the same options as in Assignment 1, and like the login menu, the options should be clickable.

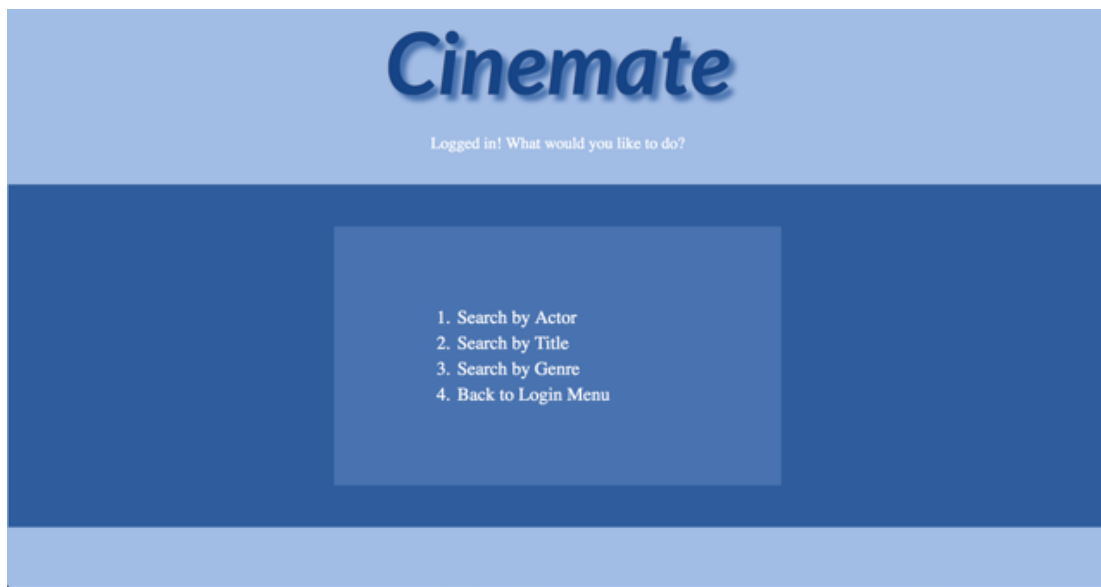


Figure 8: Movie Menu Page

If they choose to return to the login menu, they should be redirected to the login page (Figure 5).

Otherwise, all search choices should show a similar page as the screenshot below. There should be a text input as well as a search button. There should be a scrollable area underneath in which the resulting movie titles are displayed. Above the search bar should display a message that conveys whether the search is by movie title, genre or actor (in the example below, the search is by title).



Figure 9: Search Page

In addition, the user should be able to enter a new input into the search bar and be able to search again, with new results showing.

From the login page, if the user chooses to search for another user, they should be directed to the same search page as the movie search (Figure 9), but again with a changed description of the search at the top (i.e. “Enter a username or name to search by”), and the results will of course list usernames rather than movie titles.

Design Requirements

To review, here are the general aesthetic requirements for your front end:

- You must use a css file(s)
- Each page must have custom color, font, and border choices
- A search result must highlight when the mouse hovers over it (see Figure 9)
- A menu option, for both the login menu and movie menu, must highlight when the mouse hovers over it (see Figure 5)

It may be prudent to test your designs on Chrome rather than relying on Eclipse’s rendering. They will not look exactly the same, and we will be grading your assignment using a browser (versus Eclipse’s interface). Lastly, in order to make grading more straightforward for your CPs, you must name your entry page jsp file with the following convention:
'entry_<usc_username>.jsp'

Grading Criteria Total (4.5%)

Data Display: 1.0%

- 0.2% The data displayed on the feed page is accurate and complete
- 0.2% The data displayed on the profile page is accurate and complete
- 0.5% The data displayed during a search is accurate and complete
- 0.1% Valid and invalid credentials are properly distinguished

Page Navigation (1.4%)

- 0.1% A valid xml input will navigate from the entry page to the start page.
- 0.1% An invalid xml input will navigate / stay on the entry page
- 0.1% A set of valid credentials will navigate from the start page to the login page.
- 0.1% A set of invalid credentials will navigate / stay on the start page.
- 0.6% Each menu choice on the login page navigates to the correct subpage.
- 0.4% Each menu choice on the movie menu page navigates to the search page with the correct descriptive message (or back to the login page)

Page Aesthetics (1.1%)

- 0.1% At least one css file is used
- 0.1% Each page utilizes chosen colors, fonts and borders
- 0.1% Menu options highlight when the mouse hovers over them
- 0.1% Search results highlight when the mouse hovers over them
- 0.6% Each page is structured similarly to the correlated screenshots
- 0.1% Error messages are displayed on the entry and start pages when invalid inputs are provided

Other (1.0%)

- 0.1% The entry page jsp file is named as requested
- 0.2% The user search now supports searching by first and last name
- 0.1% The profile picture URL is now parsed from the xml file
- 0.5% All xml errors listed in the instructions are properly caught
- 0.1% Parsing errors displayed on the html are specific when describing the issue